FrameIt: Ontology Discovery for Noisy User-Generated Text

Dan Iter
Stanford University
daniter@stanford.edu

Alon Halevy
Megagon Labs
alon@megagon.ai

Wang-Chiew Tan
Megagon Labs
wangchiew@megagon.ai

Abstract

A common need of NLP applications is to extract structured data from text corpora in order to perform analytics or trigger an appropriate action. The ontology defining the structure is typically application dependent and in many cases it is not known a priori. We describe the FRAMEIT System that provides a workflow for (1) quickly discovering an ontology to model a text corpus and (2) learning an SRL model that extracts the instances of the ontology from sentences in the corpus. FRAMEIT exploits data that is obtained in the ontology discovery phase as weak supervision data to bootstrap the SRL model and then enables the user to refine the model with active learning. We present empirical results and qualitative analysis of the performance of FRAMEIT on three corpora of noisy user-generated text.

1 Introduction

A common task of natural language processing is to map text into structured data. The ontology of the structured data is application dependent and often represented as a set of frames with slots. Once the data is in structured form, several operations are enabled, such as performing fine-grained querying and analytics on a text corpus, or triggering responses to user utterances based on their semantics in conversational interfaces.

Existing work on mapping text to structured representations falls into two main categories: semantic role labeling (SRL) and event extraction. Research on role labeling maps text into frames of existing ontologies such as FrameNet (Baker et al., 1998) and PropBank (Palmer et al., 2005). However, these *linguistic* frame systems were designed to capture aspects of language but not specific semantics of applications. Research on event extraction tries to assemble information about events or sequences of events from multiple sentences in a document (e.g., Ahn (2006); Ji and Grishman (2008); Li et al. (2013), to name a few). In both bodies of work, much of the work concerns developing machine learning models for existing ontologies and collections of training data (He et al., 2017; Feng et al., 2016).

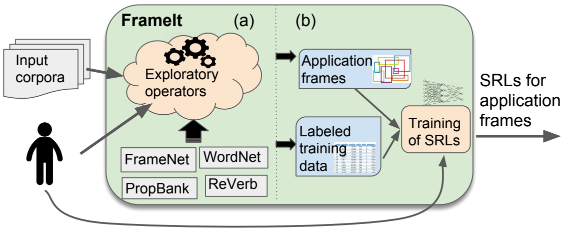

Figure 1: FRAMEIT supports the end-to-end extraction process beginning from discovery of application frames in a text corpus to training extractors for those frames. In contrast, previous research considers only part (b) in the figure where the frames are known and given in advance.

In this paper we consider an extraction setting in which the user is given a corpus of user-generated text and her goal is to discover application specific frames that will capture most of the content of the text and then train extractors for those frames. Examples of such corpora include customer reviews, free text responses to survey questions, and short personal journal entries. We describe FRAMEIT, an extraction system to support the entire process from ontology discovery to SRL training. As depicted in Figure 1, FRAMEIT differs from previous extraction work in that (1) the frames we seek do not exist in previous ontologies, and (2) the frames are not known to the user in advance.

As an example, a user of FRAMEIT may browse a corpus of descriptions of happy moments (Asai et al., 2018) with the goal of recognizing commonly occurring moments and developing extractors for these happy moments. These extractors,

Proceedings of the 2018 EMNLP Workshop W-NUT: The 4th Workshop on Noisy User-generated Text, pages 173–183
Brussels, Belgium, Nov 1, 2018. ©2018 Association for Computational Linguistics

in turn, can support a smart journal that responds intelligently or asks a useful follow-up question when such moments are entered. Upon browsing, the user notes that having meals with loved ones is a common happy experience. The user would then define a MEAL frame with attributes PARTICIPANTS, MEAL, and LOCATION, and then she would train a SRL (semantic role labeler) to recognize sentences that can be mapped into this frame. Upon further browsing, the user may conclude that many moments also mention which food was consumed during the meal, and decide to add another slot to the frame. To exemplify, Table 1 contrasts a frame defined in FRAMEIT with the frames triggered by FrameNet, PropBank and Reverb.

The contributions of this paper are: We describe the FRAMEIT System (Section 3) and how it seamlessly facilitates the exploration of the corpus, identifying and defining frames, and finally training the SRL models using a combination of weak supervision and active learning. We then evaluate FRAMEIT from two perspectives (Section 4). First, we show a result of independent interest, which is that there is a significant gap between application frames and linguistic frames of FrameNet, therefore justifying the design of FRAMEIT. Second, we demonstrate the effectiveness of FRAMEIT in defining frames for three datasets. We show that with modest effort we can create frames that cover over 70% of the sentences in two datasets and 60% on a third and achieve F1 scores near .70 on all three.

2 Related work

Work on information extraction attempts to find instances of certain predicates (e.g., CEO or MARRIEDTO), or in the case of open information extraction systems, the goal is to extract instances of *any* predicate. However, with the exception of recent work on extracting complex relations from text (Ernst et al., 2018), information extraction has focused on extracting binary relations. FRAMEIT is also similar in spirit to Ratner et al. (2017), in that it is a system for quickly creating annotations for a dataset. However, rather than on modeling labeling function interactions, FRAMEIT is more focused on the domain where the structure is not known a priori. The frames we extract with FRAMEIT also target more complex structures that can be viewed as sets of triplets.

The SRL component in FRAMEIT is reminiscent of systems for recognizing one of several intents from a user's utterance and extracting the slot values of these intents (Liu and Lane, 2016; Mesnil et al., 2015; Adel et al., 2016) (TAC KBP focuses on the latter (Roth et al., 2013; Angeli et al., 2014; Malon et al., 2012)). These slot-filling systems tend to be in very restricted domains in which the domain and the slots are known in advance. Their main goal is to extract enough values from the utterance in order to query an underlying database. In contrast, in FRAMEIT we do not know the frames in advance and an utterance may even be relevant to several frames.

There has been quite a bit of work on semantic role labeling. Unlike SRLs that map text to logical forms (Wang et al., 2015; Herzig and Berant, 2018) or focus primarily on specific linguistic structures such as predicates (He et al., 2017), FRAMEIT's SRL (like Collobert et al. (2011); Gangemi et al. (2017)) trains a neural semantic parser directly from labeled text data and maps the output to application frames that are defined by the user. Gangemi et al. (2017), like many other SRLs, is not domain-specific. Furthermore, FRAMEIT's SRL can be extended to leverage features of other SRLs such as extracted sets of named entities and locations. FRAMEIT's SRL falls into the "shallow semantic analysis" category mentioned by Abend and Rappoport (2017). It maps sentences to frame structures. In terms of Frame formalisms, our frames are consistent with the notion of semantic frames defined by Fillmore et al. (1982). However, instead of requiring the structure to be defined before the mapping is learned, FRAMEIT defines the structure simultaneously while learning the mapping.

3 The FRAMEIT System

This section describes the main features of FRAMEIT and the workflow it supports.

Problem definition: Given a text corpus, the goal of FRAMEIT is to enable a user to discover and define a set of frames that capture the contents of the text corpus and to train a SRL for each frame.

A frame is a representation of structured data. Formally, a frame is defined by its name and a set of slots (a.k.a. attributes). Slots capture spans of the text. Some attributes of a frame may have multiple values (e.g., participants in a meal). An instance of a frame may have missing values for some slots in case they are not mentioned in the text or could not be extracted reliably.

FRAMEIT is designed for the scenario in which

Sentence: I bought my mother a expensive phone for her birthday.
FrameIt Frame : **Gifts** *Gift*: phone, *Giver*: I, *Receiver*: my mother, *Occasion*: birthday
FrameNet Frame: **Commerce_buy** *Buyer*: I, *Goods*: my mother, *text*: bought FrameNet Frame: **Contacting** *text*: phone FrameNet Frame: **Expensiveness** *Goods*: phone, *text*: expensive FrameNet Frame: **Kinship** *Alter*: mother, *Ego*: my, *text*: mother FrameNet Frame: **Source_of_getting** *text*: birthday
PropBank Frame: **bought** *A0*: I, *A1*: a expensive phone, *A2*: my mother *AM-TMP*: for her birthday, *V*: bought
ReVerb Relation: **buy** *arg1*: I, *rel*: buy , *arg2*: my mother

Table 1: Frames from FrameIt, FrameNet, PropBank and ReVerb for one happy moment from HappyDB.

Listening to a podcast I love made me happy today. *My daughter offered to make dinner with me.* *My son showed me a picture he drew!* *A couple days ago I went to get ice cream, and I was happy because I haven't had ice cream in a long time.*

Table 2: Examples of happy moments from HappyDB.

the user may be knowledgeable about the domain of the corpus but not about its *content*. For example, the corpus may be a set of reviews of a product, but the user will not know which aspects of the product will be mentioned. Therefore, as shown in Figure 1(a), FrameIt's exploration support will help the user decide which frames are worth defining and what their slots should be. The goal of using FrameIt is not necessarily to capture the *entire* corpus with frames, as some of the contents may appear too infrequently to justify the effort or may be too difficult to extract or simply not sufficiently important. Note that the frames defined in FrameIt are designed for a particular application (in the same way a database schema is designed), and are different than frames in systems such as FrameNet or PropBank that are based on linguistic constructs. Section 4.1 goes into the details of comparing these two kinds of frames.

Running example: We use the HappyDB corpus (Asai et al., 2018) throughout the paper to illustrate the motivation for FrameIt and its concepts (see Table 2). HappyDB is a data set of 100K replies to the question: *describe something that made you happy in the last 24 hours* (or 3 months) collected from Mechanical Turk. Suppose we wish to build an application in which users record their significant experiences. If we could extract the essence of each experience into a structured representation, such an application can provide the user several benefits such as: (1) a dashboard that enables them to reflect on their experiences, (2) a relevant follow up question when they record an experience, or (3) provide specific advice, such as an activity that is similar to one that made them happy in the past.

Most happy experiences tend to fall into recognizable categories (Lyubomirsky, 2008). The goal of applying FrameIt to HappyDB is to discover these categories of activities, and to train extractors that recognize them in the multitude of linguistic variations in which they are expressed in the corpus and beyond.

System workflow: Working with FrameIt involves two phases that can be repeated: exploring the corpus to identify frames that capture the data to be extracted (Figure 1(a)), and training the SRL for the defined frames (Figure 1(b)). At any given point, the user may decide to resume exploration for a new frame, to refine an existing frame, or to improve the performance of the SRL by providing it better training data.

FrameIt is developed in Python and currently supports the workflow in the Jupyter notebook environment. We now describe the two phases. In our discussion we refer to the items of the corpus as sentences.

3.1 Exploring the corpus

FrameIt helps a user systematically explore a corpus, effectively discovering and defining a set of frames while simultaneously building training datasets for these frames. To motivate FrameIt's exploration features, it is important to mention the variety of goals it tries to support. These steps are common in ontology building. There are many parallels between these basic steps and those described in Noy et al. (2001).

(1) Discovery: find common patterns in the data that should be captured with frames and decide which slots these frames should have. For example, in HappyDB we might find that dining with loved ones is a common activity frequently mentioned in the corpus.

(2) Determining frame granularity: for example, instead of a frame modeling having meals with family members, we may consider a frame modeling having any social interaction with family, or a more specific frame such as having a holiday meal with family.

(3) Detecting common para-phrasings: exploring the corpus will ultimately result in creating training examples for the SRL, and therefore we should capture common para-phrasings of the concept that the frame is supposed to capture. For example, some sentences may mention *having* a meal, but others might phrase it as *making* a meal, or *cooking* a meal. Seeing different para-phrases also informs the decision about frame granularity.

(4) Creating slot dictionaries: The FRAMEIT SRL uses dictionaries of values for slots (e.g., names of meals, relatives). These dictionaries need not be perfect but it is important to bootstrap them with a good set of seeds.

Exploration features: FRAMEIT supports the discovery goal with three simple operations: (1) find a random sentence in the corpus, (2) find all the sentences that include a particular keyword or lemma (or set of keywords/lemmas), and (3) find the most commonly occurring structures (e.g., lemmas, or hypernyms, or linguistic frames) in a set of sentences.

For (3), ranking features of a set of sentences by raw counts often returns many generic features (ie. the frame and hypernym equivalent of stopwords). Instead, we sort the common structures by the rank score defined in Equation 1 to weigh each structure by its specificity to a set of sentences. Here, x denotes a structure and $c(x)$ and $C(x)$ are the counts of the structure in a subset of sentences and in the corpus, respectively. N and n are the number of sentences in the corpus and in the subset, respectively.

$$rank\ score = \frac{c(x)^2}{n} * \frac{N}{C(x)} \tag{1}$$

The next two features support the goals of granularity and para-phrasing:

Nearby sentences: find the n-nearest sentences to a given sentence. This feature finds small variations on a given sentence and could expose the need for additional slots in the frame definition or additional instances of slot values. FRAMEIT computes sentence similarity using the cosine similarity of the sentence embeddings of each sentence. Different sentence embeddings can be used, but FRAMEIT uses the mean of the word embeddings as the default.

Map to existing frame systems: Here FRAMEIT leverages other semantic tools to find different phrases that map to the same semantic category. For example, finding all the FrameNet or PropBank frames evoked by a given sentence, or the frames that are frequently evoked by a set of sentences.

FRAMEIT supports the dictionary creation goal using WordNet (Fellbaum, 1998). Specifically, given a word, FRAMEIT can find all the words in the corpus that are WordNet-siblings (or cousins, etc.) of the word. For example, "dinner", "lunch" and "breakfast" all share the hypernym "meal". This set can be expanded by including all other terms in the corpus for which "meal" is a hypernym, including infrequent terms such as "potluck", "luncheon" and "seder".

Example 3.1 *A user can easily discover that "dinner" is mentioned often in* HAPPYDB *by looking at the most frequent lemmas in the corpus. Looking at the sentences most similar to those containing "dinner", the user finds that dining experiences are often described with a set of attributes including the specific food, an adjective (e.g., delicious), when the meal took place and other participants.*

The user can also explore the most common FrameNet frames in the set of happy moments containing dinner. For example, we find the "food" FrameNet frame gets evoked on all food names and the social_event frame gets triggered on gatherings such as dinner and parties. The latter FrameNet frame may suggest additional slots (e.g., occasion) to the definition of the dining frame. Furthermore, one can also exploit FrameNet frames to determine the set of sentences that are relevant as training data. For example, all sentences that evoked the food or social_event frame may be included as part of the training data for the dining frame.

To support interactive exploration FRAMEIT pre-processes the corpus by creating an index on the words and lemmas in the corpus. Additionally, FRAMEIT runs Sempahor (Das et al., 2014) to trigger the frames in FrameNet and runs an SRL described by He et al. (2017) to map each sentence to PropBank frames.

3.1.1 Defining Frames

After exploration, the user specifies a frame by defining its name and slots. For example, the user can create a frame named MEALS and add slots for PARTICIPANT, MEAL and FOOD. The ranges of the slots can be defined by appropriate dictionaries (e.g., a list of meals). Alternatively, we can attach a recognizer, such as an off-the-shelf pretrained text extractor, for the range of an attribute.

In addition to typing of slots, the user can specify a hierarchy on frames and on slot domains and enjoy the benefits of inheritance. For example, the user can specify that MEALSWITHFAMILY is a subclass of MEALS and that slot MEAL has a superset of the values of the slot HOLIDAYMEAL.

3.2 Training the SRL

The end goal of using FRAMEIT is to define an ontology and obtain an SRL model that can map text to those defined frames. While the user is exploring the corpus, they are simultaneously generating a training dataset for their SRL. FRAMEIT supports a two-phase approach to training the SRL. In the first phase, the user provides a set of possibly noisy training data as weak supervision to bootstrap the model. The training data is created as a natural side effect of exploring the corpus. In the second phase, FRAMEIT uses active learning to improve the SRL model.

3.2.1 Bootstrapping with weak supervision

Weak supervision refers to a setting where a model is trained using "noisy" labels or labels from a different context (Mintz et al., 2009; Wu and Weld, 2010; Fader et al., 2011; Sa et al., 2016; Ratner et al., 2016, 2017; Craven et al., 1999; Androutsopoulos and Malakasiotis, 2010). In FRAMEIT, the "different context" is external data and automatic annotations provided by ontologies such as FrameNet and WordNet. Specifically, as the user explores the corpus, she uses the FRAMEIT operators to explore sets of sentences that describe concepts that should be classified under a particular frame. These sets can then readily be used as seed sets for training.

As a natural byproduct of the exploration, these sets of sentences contain different linguistic expressions of the data that should be captured by the frame. In our example, the user can create a set of sentences that have a "meal" term, have triggered the FrameNet "Food" frame and that mention a person. This set will be the set of examples for the MEALS frame. After being given positive training examples, FRAMEIT automatically samples the corpus for negative training data and splits the training data into a training and validation set to monitor overfitting.

Each frame has a binary classifier, which is implemented with a 3-layer convolutional neural network followed by two fully connected layers, similar to the one described by Kim (2014). The input is a dense matrix $D^{k \times n}$ where k is the number of words in the sentence and n is the size of the word embeddings. All convolutional filters match the word embedding dimension. At inference time, a sentence is input into the binary classifier of each frame in parallel.

Once a sentence has been classified to contain a given frame, we run a binary classifier for each slot of the frame. As noted earlier, the user may provide a dictionary or a recognizer for a slot, and may constrain a slot to be a certain part of speech.

We distinguish between two types of models for frame slots. The first type of model is context *independent*. The slot FOOD of the MEALS frame is context independent, and therefore we use a linear regression model that predicts if a word is a food or not. Interestingly, this simple method for word set expansion works surprisingly well. The second kind is context *dependent*, which means that the meaning of the word is dependent on the context of the sentence. For example, whether a person is the one providing a gift or receiving a gift is dependent on the context of the sentence. For these slots we embed the entire sentence using the same architecture as we used for the frame classifier and then concatenate the sentence embedding with the word embedding of the candidate slot value. Finally, we apply a fully connected layer and output the binary prediction. Empirically, we found that simple models with correct regularization are sufficient for the task of extracting most sentences in a corpus that express a well defined frame, which we show in Section 4.2 (SRL performance).

3.2.2 Model refinement with Active learning

FRAMEIT provides an active learning interface to help the user debug the SRL model.

After distant supervision rules have been applied to generate a seed set and an initial SRL model has been trained, the user has access to the noisy labels created by the rules and the initial SRL labels on the entire corpus. The model can be improved by improving the training data, for which there are two simple strategies: (1) adding more data to the training set, (2) fixing incorrect labels in the seed set. For (1) the seed set can be expanded by including previously unlabeled examples that have high confidence positive labels from the initial SRL model. For (2), we can use the confidence of the SRL model on training data to find examples that may be false positives or false negatives. In both cases, we have the user label the sampled examples before updating the training data. Another common technique in active learning is uncertainty sampling (Lewis and Gale, 1994). This strategy can be used in addition to the ones above to find and label challenging examples and potentially improve the SRL model on examples that are at the boundary.

The choice of strategy and number of labeled examples is a parameter set by the user. Selecting one strategy and labeling k examples is one iteration of active learning. The output of each iteration is a new training dataset with labels, which augments the previous dataset according to the labels provided by the user and which can be used to retrain the SRL. Additionally, the user may choose to update rules used to create the training data. For example, a common error in the parser that we noticed is that indirect objects are labeled as direct objects. During the first iteration of active learning of the Bought-object attribute of the Buying frame, we noticed many positive examples of person names. By updating our rules to filter out people entities, we were able to quickly increase the precision of the model in only one iteration.

4 Experiments

Here, we show that application frames are qualitatively different from linguistically inspired frames thereby justifying the fact that FrameIt extracts them directly from data. We also experimentally evaluate the different components of FrameIt.

4.1 Application vs. linguistic frames

We establish that the gap between existing frame systems, such as FrameNet, PropBank, and VerbNet and FrameIt can be quite large as the former are meant to capture linguistic concepts while FrameIt is meant to capture application specific concepts. For space considerations, we focus on the problem of classifying a sentence into a FrameIt frame and not on extracting attributes. We empirically show that the gap exists by showing (1) that any set of sentences will map to a huge number of unique linguistic frames, many of which are not relevant for an application, (2) naively using the most common linguistic frames in a set of example sentences may be good for high coverage but leads to lower precision and (3) in some cases, FrameNet frames are not even useful as features in a fully supervised classification task.

	Meal	Promotion	Buying
Frame Examples	11860	1677	3019
Frames triggered in other systems			
FrameNet	595	378	474
FrameNet (Attr)	1957	1092	1397
PropBank	1646	512	750
ReVerb	2709	537	951
Per Sentence Stats Average			
FrameNet	5.48	5.36	5.76
FrameNet (Attr)	12.10	12.05	13.45
PropBank	1.9	2.9	2.15
ReVerb	1.12	1.14	1.25

Table 3: The first row shows the number of sentences in HappyDB extracted by FrameIt for three frames. The second set of rows shows the number of frames in other systems that are triggered by these sets of sentences. The last section shows the average number of frames triggered per sentence.

We consider the sets of sentences in HappyDB that FrameIt classified into the Meals, Promotions and Buying frames[1]. We denote these sets of sentences by S_1, S_2, and S_3 respectively. Table 3 shows that S_1, S_2, and S_3 triggered 595, 378, and 474 unique frames in FrameNet, respectively, and the numbers for PropBank and Reverb are even higher. The mappings are produced by the SRLs provided by (Das et al., 2014; He et al., 2017; Fader et al., 2011) respectively. Furthermore, they populated 10135, 1446, 2536 attributes in these frames, respectively. These numbers suggest that even though the sets of sentences of a particular FrameIt frame refer to the same general concept, they tend to map to many diverse FrameNet frames. Hence, trying to define a FrameIt frame in terms of other frames would be tedious at best, even ignoring the need to be an expert on the contents of other frame systems.

It could, of course, be the case that most of the FrameNet frames are unimportant. Perhaps a FrameIt frame can be expressed as the disjunction of a small number of FrameNet frames. Specifically, for each $k, 1 \leq k \leq 15$, we considered the set $F_{i,k}$ of most frequently triggered frames in S_i, and we computed the set of sentences in HappyDB that would trigger any frame in $F_{i,k}$, which we denote by $H_{i,k}$. Figure 2 shows the precision and recall of $H_{i,k}$ w.r.t. S_i for the Meal frame. Even though very high recall can be achieved, the precision quickly decreases because FrameNet frames are often very general.

[1] We obtain similar results for other frames although we do not show them here.

The same results were obtained for the other two frames and for a broader set of propositional formulas over FrameNet frames (including conjunctions and negation), but we omit the details for space considerations.

We also ask if the combination of a sentence's FrameNet frames (as opposed to words) is sufficient to identify that it belongs to a FrameIt frame. We test this hypothesis with a Logistic Regression model developed as follows. We consider the sets S_1, S_2 and S_3 as the ground truth for the frames mentioned above. While they are noisy, Table 6 shows that the model used to generate these sets has reasonably high accuracy on a held out test set. We represent each sentence by a binary vector, where each index maps to a FrameNet frame, labeled as 1 if the frame is annotated on that sentence. To limit the number of features used, we only use those that appear in the ground truth positive examples (the FrameNet row in Table 3). We fit a Logistic Regression model to the ground truth data and an equal number of random samples from the rest of the corpus. The goal of the classifier is to predict the FRAMEIT frame from this representation. The F1 scores for each model are 0.722 for MEALS, 0.813 for BUYING and only 0.235 for PROMOTION. As expected, FRAMEIT frames can be modeled with FrameNet frames to the extent that FrameNet contains relevant frames. In the case of BUYING, it is nearly a one-to-one mapping. However, since PROMOTION is not represented unambiguously by any FrameNet frame, no combination of FrameNet frames will sufficiently represent this FRAMEIT frame.

The purpose of these explorations is to demonstrate some of the challenges of finding good frame representations and demonstrating that relying solely on linguistic frames may not be sufficient for some applications. In summary, we show that with linguistic frames, we can either achieve high recall but low precision or high precision or low recall. It is tedious to use linguistic frames to express FRAMEIT frames and they are also not a good feature for representing FRAMEIT frames in general.

4.2 Evaluating FRAMEIT components

We first show that FRAMEIT is a useful tool to capture the salient aspects of different corpora. We then show the performance of the SRL of FRAMEIT and the additional improvements obtained with active learning.

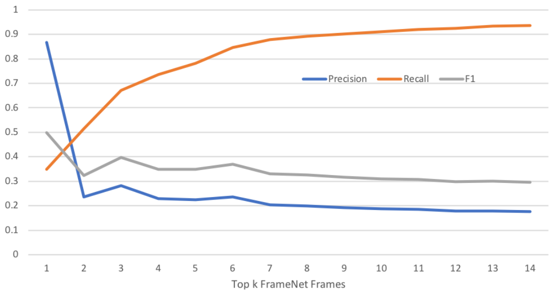

Figure 2: Disjunctions of FrameNet frames for expressing the "Meal" frame.

We evaluate FRAMEIT on three datasets: (1) **HappyDB** (Asai et al., 2018), described earlier. (2) **TripAdvisor hotel reviews** (Wang et al., 2011). We subsample the dataset down to 78K reviews of hotels on the TripAdvisor website. We do not use the associated ratings provided but do note that they are a hint to what are some common aspects of the data (room, location, service, etc). (3) **ANES 2008 presidential election survey** (DeBell et al., 2010). A survey that concluded with a free form response from which we extracted 2K sentences of responses.

FRAMEIT is evaluated on 3 datasets, all composed of short user-generated texts. FRAMEIT is designed to work on short texts, where there are no long term dependencies or overarching themes or concepts. The 3 datasets vary in their domains and the extent to which sentences in the corpora map well to linguistic frames.

Corpus coverage Next, we show that with a modest amount of work, the FRAMEIT workflow enables us to capture the parts of the corpora that can be extracted into meaningful frames. We define the *coverage* of a set of frames as the percentage of sentences in the corpus that trigger at least one frame. Note that coverage is not recall because there is no ground truth of frames for each sentence. We report this metric as an estimate of how complete our ontology is with respect to the corpus. For each dataset, we create frames until we can no longer define a new meaningful application frame that would cover at least 1% of the unframed sentences in the corpus. To find new frames, we consider the most common FrameNet and Propbank frames and Reverb extractions among the unframed sentences to see if there are any good candidates for FRAMEIT frames.

Our results are shown in Table 4. For HAPPYDB and ANES2008 we reach around 70% coverage while TripAdvisor we only get to 62% cov-

	HappyDB	TripAdvisor	ANES2008
# frames	19	13	12
Coverage	70%	62%	71%
F1	0.766	0.796	.742
Prec./Rec.	0.72/0.82	0.76/0.84	0.65/0.86

Table 4: Percentage of sentences covered by frames on the three datasets. Each created frame covers at least 1% of the corpus. F1 score is computed on a hold out set of 100 examples. The scores are averaged over all the frames defined.

erage because we split multi-sentence reviews into single sentences, leaving some sentences meaningless without context. We also report F1 scores for the SRL on a holdout set of 100 sentences that are manually labeled based on the created frames. The *precision* measures the percentage of sentences that trigger the correct frame and the *recall* is the percentage of examples of each frame that are correctly classified by the SRL. All metrics are averaged over all frames weighted by how often the frame appears. For example, if one sentences is labeled with 3 frames but our SRL emits 2 frames, one of which is incorrect, this will result in a precision of .5 and recall of .33.

SRL performance This section evaluates FrameIt's SRL. For the purpose of evaluation, we manually labeled 100 examples for each frame (half positive and half negative), omit these examples from the training data and use those examples as the test set. Note that these 100 examples are not the same as those used in for Corpus coverage above. We choose eight frames and six attributes for which to create ground truth labeled data for evaluation. In this section, we present F1 scores to demonstrate empirically that FrameIt can learn a high accuracy model that does not overfit the labeled data. Generally, the performance of the SRL is highly dependent on the quality of weak supervision rules. All the frames evaluated in this section use simple rules similar to those in Table 5.

The left side of Table 6 reports the F1 scores for identifying the correct frame and the right reports the results for extracting the attributes. We note that frames vary quite a bit in their scope. For example, the Meals frame represents any sentence of a person having any kind of meal, which is very broad. Alternatively, the Seeing someone frame is constrained to seeing or spending time with another person as opposed to a movie or event. Furthermore, some frames are closer to linguistic frames (e.g., Buying is similar to a FrameNet frame but also includes *purchase* and *get*). Conversely, *Exercise* is an application frame that includes all activities that might be classified as exercise including *going to a gym*, *running*, *playing basketball*, *working out*, and has no counterpart in FrameNet.

Frame	Rules
Meals	*induces FrameNet frame "Food" OR contains a word with the hypernym "meal"*
Promotion	*(contains lemma "promotion" OR "promote") OR ((contains lemma "raise" OR "bonus") AND (mentions "job" OR "work" OR "boss"))*
Buying	*contains the lemma "buy"*

Table 5: The weak supervision rules used to find high precision examples for each FrameIt frame.

For the attributes, recall that FrameIt provides two types models for attributes; (1) logistic regression on word sets that is context independent and (2) neural networks including sentence context. The context-based models can represent more complex attributes than the logistic regression model but is more likely to overfit the training data. For example, the Meals attribute is perfect on the test data because there is a small set of meal terms while the Buying-object attribute must correctly extract object that may not be the direct object of the verb, such as "we saw a **house** we loved so we bought *it*".

Human-in-the-loop Effort FrameIt is not an automatic or end-to-end system and therefore a human user plays a critical role. It is challenging to quantify human effort for the ontology discovery task but we can provide some simple statistics about how much time and code was required to collect our results. A total of 44 frames were created. Most rules used to collect the initial distant supervision set were similar to those in Table 5. Rules are discovered by looking at the most common or most salient hypernyms and frames for a small seed set of examples. FrameIt indexes the corpus and all hypernyms and frames so generating these lists is instantaneous and it takes on the

Frame	F1	Attribute	Acc.
Seeing someone	0.76	Foods	0.93
Going to a location	0.79	People	0.85
Exercising	0.87	Meals	1.00
Watching something	0.93	Buying-Object*	0.87
Promotion	1.00	Buying-Buyer*	0.94
Meals	1.00	Buying-Receiver*	0.84
Buying something	1.00		
Winning	0.99		

Table 6: F1 scores on the test set for sentence level frames and attributes. Attributes with an asterisk are trained with the context and using a neural network. Others are with logistic regression.

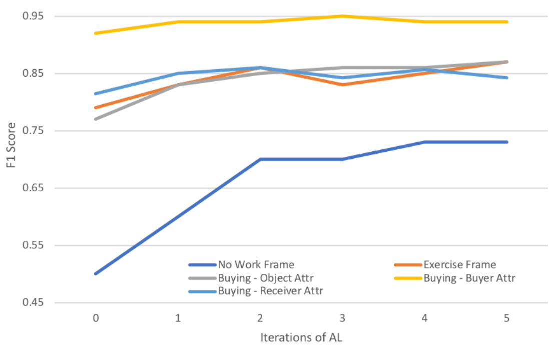

Figure 3: Model improvement with active learning. Improvement of F1 on 100 sentence hold out set for NoWork, Exercise and attributes of the Buying frame: BoughtObject, Buyer, Receiver.

order of minutes of exploration to discover a sufficient set of rules for a single FrameIt frame. Roughly 2000 lines of code were written to discover and train the 19 frames for HappyDB. As shown in Figure 3, at most 5 iterations of active learning are done on each frame. Each iteration takes about a minute of labeling and a few minutes of reviewing new labels and updating rules. The total ranges from 10-30 minutes per frame, where all of the time is spent on simple tasks that don't require expertise in linguistics or machine learning.

Active Learning We evaluated the active learning component on five models whose initial SRL results were relatively low. In each iteration, the user labels 10 examples (as described in Section 3.2.2). After each iteration the user is allowed to update a rule, such as creating a dictionary of negative or positive words. The F1 scores are evaluated on a 100 sentence holdout set.

The graph in Figure 3 shows improvements in F1 scores, ranging from 24% - 46% decrease in F1 error. Improvements come primarily from two types of corrections; (1) finding errors in the rules and (2) generalizing from rules based on entities. For example, a common mistake with the "Buying - object" attribute was that the weak supervision used the direct object of the "buy" verb, but this was often incorrectly parsed as the person for whom the gift was bought. Active learning helps quickly find a list of terms describing people for whom things are often bought (SOs, children, friends, etc) to fix the weak supervision rules. We observed that the model was also able to generalize beyond the rigged rules. For examples, the model extracts "controller" from the sentence "I fixed my Xbox one controller on my own so I didn't have to buy a new one" even though it is not the direct object of the "buy" verb. Lastly, we also observe the context dependent attribute models learn common patterns in text. The "Buying - object" attribute is only trained on "buying" related sentences, but when applied to any other sentences, it consistently extracts direct objects, despite having no access to POS tags or dependency parse tree tags in the input and having never seen some entities in the training data.

5 Conclusion and Future Work

We described the FrameIt system that provides an end-to-end workflow beginning from the exploration of a text corpus to training SRL models that map natural language text into application specific frames. In addition to empirically evaluating FrameIt, we showed that application frames are qualitatively different from linguistically inspired frames.

One of the major directions for future work is for FrameIt to support the exploration process further by taking a more active role in suggesting possible frames and different frame granularity that the user should consider. In particular, in building FrameIt we have discovered 2 primary challenges that limit the quality of the final ontology and SRL model. (1) Given a small set of sentences from a corpus, can a system automatically find other sentences that belong to the same frame but increase the diversity of the set without changing the meaning? For example, expanding a set of sentences about "dinner" to include "lunch" and "breakfast" but not other activities that can be "had" or "gotten". (2) Given a large set of sentences, can a system automatically discover all the aspects of an activity and correctly group related terms? For example, given sentences about "meals" can we automatically discover that it can be "bought" or "cooked" and "delicious" or "gross". Future FrameIt work will focus more on offloading these responsibilities from the user and moving towards more model-based generation of structure.

Acknowledgements

We thank Tom Mitchell and Dan Jurafsky for their thoughtful comments. We also thank Behzad Golshan, George Mihaila and Jonathan Engel who are active FrameIt users and provided valuable feedback.

References

Omri Abend and Ari Rappoport. 2017. The state of the art in semantic representation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 77–89.

Heike Adel, Benjamin Roth, and Hinrich Schutze. 2016. Comparing convolutional neural networks to traditional models for slot filling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

David Ahn. 2006. The stages of event extraction. In *ARTE Workshop*, pages 1–8.

Ion Androutsopoulos and Prodromos Malakasiotis. 2010. A survey of paraphrasing and textual entailment methods. *Journal of Artificial Intelligence Research*, 38:135–187.

Gabor Angeli, Sonal Gupta, Melvin Jose, Christopher D. Manning, Christopher Re, Julie Tibshirani, Jean Y. Wu, Sen Wu, , and Ce Zhang. 2014. Stanfords 2014 slot filling systems. In *TAC KBP*.

Akari Asai, Sara Evensen, Behzad Golshan, Alon Halevy, Vivian Li, Andrei Lopatenko, Daniela Stepanov, Yoshihiko Suhara, Wang-Chiew Tan, and Yinzhan Xu. 2018. Happydb: A corpus of 100,000 crowdsourced happy moments. In *Proceedings of LREC 2018*, Miyazaki, Japan. European Language Resources Association (ELRA).

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The berkeley framenet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL '98, pages 86–90, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ronan Collobert, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. In *Journal of Marchine Learning Research*, volume 12.

Mark Craven, Johan Kumlien, et al. 1999. Constructing biological knowledge bases by extracting information from text sources. In *ISMB*, volume 1999, pages 77–86.

Dipanjan Das, Desai Chen, Andr F. T. Martins, Nathan Schneider, and Noah A. Smith. 2014. Frame-semantic parsing. In *Computational Linguistics*, volume 40.

Matthew DeBell, Jon A Krosnick, and Arthur Lupia. 2010. Methodology report and users guide for the 2008–2009 anes panel study.

Patrick Ernst, Amy Siu, and Gerhard Weikum. 2018. Highlife: Higher-arity fact harvesting. In *WWW*, pages 1013–1022.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the conference on empirical methods in natural language processing*, pages 1535–1545. Association for Computational Linguistics.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

Xiaocheng Feng, Lifu Huang, Duyu Tang, Heng Ji, Bing Qin, and Ting Liu. 2016. A language-independent neural network for event detection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 66–71.

Charles J Fillmore et al. 1982. Frame semantics. *Cognitive linguistics: Basic readings*, pages 373–400.

Aldo Gangemi, Valentina Presutti, Diego Reforgiato Recupero, Andrea Giovanni Nuzzolese, Francesco Draicchio, and Misael Mongiova. 2017. Semantic web machine reading with fred. volume 8, pages 873–893.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and whats next. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

J. Herzig and J. Berant. 2018. Decoupling structure and lexicon for zero-shot semantic parsing. *arXiv preprint arXiv:1804.07918*.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *ACL*, pages 254–262.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*.

David D Lewis and William A Gale. 1994. A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–12. Springer-Verlag New York, Inc.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *ACL*, pages 73–82.

Bing Liu and Ian Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. In *Interspeech*.

Sonja Lyubomirsky. 2008. *The How of Happiness: A Scientific Approach to Getting the Life You Want*. Penguin Books.

Christopher Malon, Bing Bai, and Kazi Saidul Hasan. 2012. Slot-filling by substring extraction at tac kbp (team papelo). In *TAC KBP*.

G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. Hakkani-Tur, X. He, L. Heck, G. Tur, D. Yu, and G. Zweig. 2015. Using recurrent neural networks for slot filling in spoken language understanding. In *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP*.

Natalya F Noy, Deborah L McGuinness, et al. 2001. Ontology development 101: A guide to creating your first ontology.

Martha Palmer, Dan Gildea, and Paul Kingsbury. 2005. The proposition bank: A corpus annotated with semantic roles. In *Computational Linguistics Journal*, volume 31.

Alex Ratner, Christopher De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. 2016. Data programming: Creating large training sets, quickly.

Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. *Proceedings of the VLDB Endowment*, 11(3):269–282.

Benjamin Roth, Tassilo Barth, Michael Wiegand, Mittul Singh, , and Dietrich Klakow. 2013. Effective slot filling based on shallow distant supervision methods. In *the Sixth Text Analysis Conference (TAC 2013)*.

Christopher De Sa, Alex Ratner, Christopher Ré, Jaeho Shin, Feiran Wang, Sen Wu, and Ce Zhang. 2016. Deepdive: Declarative knowledge base construction. *SIGMOD Rec.*, 45(1):60–67.

Hongning Wang, Yue Lu, and ChengXiang Zhai. 2011. Latent aspect rating analysis without aspect keyword supervision. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 618–626. ACM.

Yushi Wang, Jonathan Berant, and Percy Liang. 2015. Building a semantic parser overnight. In *ACL*.

Fei Wu and Daniel S Weld. 2010. Open information extraction using wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 118–127. Association for Computational Linguistics.